IN THE CLAIMS

Please amend the claims as follows:

1. (Currently Amended) [[A]] In a computer-aided design and verification system, a method for instrumenting a hardware description language (HDL) design entity that is generated utilizing a HDL source code file within the syntax convention of a platform HDL, wherein said HDL source code file includes a description of at least one operating event within the conventional syntax of said platform HDL, said method comprising:

describing an instrumentation entity within said HDL source code file utilizing a non-conventional syntax comment such that said instrumentation entity is embedded within said design entity without being incorporated into an overall design in which said design entity is incorporated, wherein said describing step further comprises associating said instrumentation entity with said at least one operating event utilizing a non-conventional syntax comment within said HDL source code file.

2. (Cancelled)

3. (Currently Amended) The method of claim [[2]] 1, wherein said conventional syntax description of said at least one operating event includes a description of at least one design entity element, said associating step further comprising specifying said design entity utilizing a non-conventional syntax comment.

4. (Original) The method of claim 1, wherein said describing step further comprises declaring a name associated with said instrumentation entity utilizing a non-conventional syntax comment within said HDL source code file.

5. (Original) The method of claim 1, wherein said describing step further comprises declaring the presence of an embedded instrument within said HDL source code file utilizing a non-conventional syntax comment within said HDL source code file.

6. (Original) The method of claim 5, further comprising compiling said HDL source code file using a compiler that interprets said non-conventional syntax comment such that said compiler recognizes said declared presence of said embedded instrument.

7. (Original) The method of claim 6, further comprising detecting one or more non-conventional syntax comments indicating the presence of one or more embedded instruments during said compiling step.

8. (Original) The method of claim 7, wherein during a model build process, said compiler translates said design entity HDL source code file into a compiled proto data structure, said method further comprising setting an embedded instrument flag within said compiled proto data structure in response to detecting a non-conventional syntax comment indicating the presence of an embedded instrument.

9. (Original) The method of claim 8, wherein upon compilation of said design entity HDL source code file into said compiled proto data structure, said proto data structure is passed to an instrumentation load tool, whereupon said method further comprises determining whether said embedded instrument flag is set within said proto data structure.

10. (Original) The method of claim 9, wherein in response to determining that said embedded instrument flag is set, said method further comprises opening and parsing said design entity HDL source code file to locate one or more embedded instrumentation entity comments.

11. (Original) The method of claim 10, further comprising generating a proto data structure and an instance data structure corresponding to said instrumentation entity in accordance with said embedded instrumentation entity comments.

12. (Original) The method of claim 9, wherein upon passing said proto data structure to said instrumentation tool, said method further comprises removing said embedded instrument flag from said proto data structure.

13. (Currently Amended) [[A]] In a computer-aided design and verification system, a computer program product stored on a computer-readable medium for instrumenting a hardware description language (HDL) design entity that is generated utilizing a HDL source code file within the syntax convention of a platform HDL, wherein said HDL source code file includes a description of at least one operating event within the conventional syntax of said platform HDL, said program product comprising:

instruction means for describing an instrumentation entity within said HDL source code file utilizing a non-conventional syntax comment; ~~and~~

instruction means for translating said non-conventional syntax comment such that said instrumentation entity is embedded within said design entity without being incorporated into an overall design in which said design entity is incorporated; and

instruction means for associating said instrumentation entity with said at least one operating event utilizing a non-conventional syntax comment within said HDL source code file.

14. (Cancelled)

15. (Currently Amended) The program product of claim [[14]] 13, wherein said conventional syntax description of said at least one operating event includes a description of at least one design entity element, said program product further comprising instruction means for specifying said design entity utilizing a non-conventional syntax comment.

16. (Original) The program product of claim 13, further comprising instruction means for declaring a name associated with said instrumentation entity utilizing a non-conventional syntax comment within said HDL source code file.

17. (Original) The program product of claim 13, further comprising instruction means for declaring the presence of an embedded instrument within said HDL source code file utilizing a non-conventional syntax comment within said HDL source code file.

18. (Original) The program product of claim 17, further comprising instruction means for compiling said HDL source code file using a compiler that interprets said non-conventional

syntax comment such that said compiler recognizes said declared presence of said embedded instrument.

19. (Original) The program product of claim 18, further comprising instruction means for detecting one or more non-conventional syntax comments indicating the presence of one or more embedded instruments during said compiling.

20. (Original) The program product of claim 19, wherein during a model build process, said compiler translates said design entity HDL source code file into a compiled proto data structure, said program product further comprising instruction means for setting an embedded instrument flag within said compiled proto data structure in response to detecting a non-conventional syntax comment indicating the presence of an embedded instrument.

21. (Original) The program product of claim 20, wherein upon compilation of said design entity HDL source code file into said compiled proto data structure, said proto data structure is passed to an instrumentation load tool, whereupon said program product further comprises instruction means for determining whether said embedded instrument flag is set within said proto data structure.

22. (Original) The program product of claim 21, wherein in response to determining that said embedded instrument flag is set, said program product further comprises instruction means for opening and parsing said design entity HDL source code file to locate one or more embedded instrumentation entity comments.

23. (Original) The program product of claim 22, further comprising instruction means for generating a proto data structure and an instance data structure corresponding to said instrumentation entity in accordance with said embedded instrumentation entity comments.

24. (Original) The program product of claim 21, wherein upon passing said proto data structure to said instrumentation tool, said program product further comprises instruction means for removing said embedded instrument flag from said proto data structure.

25. (Currently Amended) [[A]] In a computer-aided design and verification system, a system for instrumenting a hardware description language (HDL) design entity that is generated utilizing a HDL source code file within the syntax convention of a platform HDL, wherein said HDL source code file includes a description of at least one operating event within the conventional syntax of said platform HDL, said system comprising:

processing means for describing an instrumentation entity within said HDL source code file utilizing a non-conventional syntax comment; and

processing means for translating said non-conventional syntax comment such that said instrumentation entity is embedded within said design entity without being incorporated into an overall design in which said design entity is incorporated; and

processing means for associating said instrumentation entity with said at least one operating event utilizing a non-conventional syntax comment within said HDL source code file.

26. (Cancelled)

27. (Currently Amended) The system of claim [[26]] 25, wherein said conventional syntax description of said at least one operating event includes a description of at least one design entity element, said system further comprising processing means for specifying said design entity utilizing a non-conventional syntax comment.

28. (Original) The system of claim 25, wherein said processing means for describing and instrumentation entity further comprises processing means for declaring a name associated with said instrumentation entity utilizing a non-conventional syntax comment within said HDL source code file.

29. (Original) The system of claim 25, wherein said processing means for describing an instrumentation entity further comprises processing means for declaring the presence of an embedded instrument within said HDL source code file utilizing a non-conventional syntax comment within said HDL source code file.

30. (Original) The system of claim 29, further comprising a compiler that interprets said non-conventional syntax comment such that said compiler recognizes said declared presence of said embedded instrument within said design entity.

31. (Original) The system of claim 30, further comprising processing means for detecting one or more non-conventional syntax comments indicating the presence of one or more embedded instruments during compilation of said design entity.

32. (Original) The system of claim 31, wherein during a model build process, said compiler translates said design entity HDL source code file into a compiled proto data structure, said system further comprising processing means for setting an embedded instrument flag within said compiled proto data structure in response to detecting a non-conventional syntax comment indicating the presence of an embedded instrument.

33. (Original) The system of claim 32, wherein upon compilation of said design entity HDL source code file into said compiled proto data structure, said proto data structure is passed to an instrumentation load tool, whereupon said system further comprises processing means for determining whether said embedded instrument flag is set within said proto data structure.

34. (Original) The system of claim 33, wherein in response to determining that said embedded instrument flag is set, said system further comprises processing means for opening and parsing said design entity HDL source code file to locate one or more embedded instrumentation entity comments.

35. (Original) The system of claim 34, further comprising processing means for generating a proto data structure and an instance data structure corresponding to said instrumentation entity in accordance with said embedded instrumentation entity comments.

36. (Original) The system of claim 33, wherein upon passing said proto data structure to said instrumentation tool, said system further comprises processing means for removing said embedded instrument flag from said proto data structure.